2000

Software Mode Changes for Continuous Motion Tracking

E. G. Araujo
University of Massachusetts Amherst

Software Mode Changes for Continuous Motion Tracking*

E. G. Araujo, D. R. Karuppiah, Y. Yang, R. A. Grupen
P. A. Deegan, B. S. Lerner, E. M. Riseman, Z. Zhu

Department of Computer Science
University of Massachusetts, Amherst, MA 01003 USA

Abstract

Robot control in nonlinear and nonstationary run-time environments presents challenges to traditional software methodologies. In particular, robot systems in "open" domains can only be modeled probabilistically and must rely on run-time feedback to detect whether hardware/software configurations are adequate. Modifications must be effected while guaranteeing critical performance properties. Moreover, in multi-robot systems, there are typically many ways in which to compensate for inadequate performance. The computational complexity of high dimensional sensorimotor systems prohibits the use of many traditional centralized methodologies.

We present an application in which a redundant sensor array, distributed spatially over an office-like environment can be used to track and localize a human being while reacting at run-time to various kinds of faults, including: hardware failure, inadequate sensor geometries, occlusion, and bandwidth limitations. Responding at run-time requires a combination of knowledge regarding the physical sensorimotor device, its use in coordinated sensing operations, and high-level process descriptions. We present a distributed control architecture in which run-time behavior is both preanalyzed and recovered empirically to inform local scheduling agents that commit resources autonomously subject to process control specifications. Examples will be available from our search and rescue platform[1].

1 Introduction

High-level deliberation and low-level reactivity are valuable in the control of autonomous and self-adaptive systems. A successful implementation of such a hybrid architecture would permit the system to make use of prior knowledge when appropriate and to respond quickly to run-time data. The central open question appears to be deciding how reacting and deliberating should interact in a constructive fashion. We have adopted a perspective in which the control hierarchy is adaptive at every level. Low-level control processes parameterized by resources interact with the domain continuously and recover context observable by the working set of control. This kind of context feedback can be used to identify the current run-time environment and permits the high-level process planner to re-deploy resources so as to address the goals of the system and to change the kinds of context feedback available. Over time, robust plans for interacting with specific problem domains are compiled these policies into rich, comprehensive reactive policies. State descriptions evolve to express likely run-time context at the highest levels and reactive policies adapt to handle run-time contingencies at the lowest levels.

We are concentrating on how sensory and computational resources, distributed in a non-uniform manner over multiple mobile platforms can be coordinated to achieve mission objectives. Our approach relies on technologies that produce flexibility, resourcefulness, high performance, and fault tolerance. Specifically, we are interested in (1) how cross-modal sensory front-ends can be designed to provide mission-specific percepts, (2) how perceptual behavior can incorporate sensory information derived from two or more robotic platforms carrying different sensors and feature extraction algorithms, and (3) how team resources can be organized effectively and how low-level sensory and motor activity can be scheduled to achieve multiple simultaneous objectives.

A family of resource scheduling policies, called Behavior Programs (B-Pgms), is downloaded into each member of a working group of robots as part of the configuration process. Each B-Pgm contains a set of (previously evaluated) contingency plans with which to respond to a variety of likely run-time contexts. This policy is responsible for orchestrating the run-time behavior of the system in response to percepts gathered on-line. The temporal history of states produced by a particular B-Pgm defines a run-time

*This work was supported by AFRL/IFTD under F30602-97-2-0032 (SAFER), DARPA/ITO DABT63-99-1-0022 (SDR Multi-Robot), and NSF CDA-9703217 (Infrastructure).

[1] http://www-robotics.cs.umass.edu/rob_safer

context and supports probabilistic performance predictions for the team. These predictions are continuously refined and updated for use in higher-level planning. Run-time contexts that have may be handled by making use of contingency plans in the B-Pgm, by re-deploying resources, or by replanning at the mission/process planning level if required.

The UMass hybrid architecture is based on a set of primitive, closed-loop control processes. This framework allows hierarchical composition of the controllers into behavior programs (B-Pgms) for tracking, recognition, motion control, and for a more complex human tracking scenario. We are developing schemes for automatically programming behavior in a variety of meaningful contexts and subsequently using these policies as abstract actions in a growing Semi-Markov Decision Process (SMDP). These hierarchically organized processes are implemented in a distributed, real-time environment in which we are developing mechanisms for multi-threaded behavior. Moreover, the multi-robot platform is designed to respond to multiple, simultaneous objectives and reasons about resources using a high-level process description and control procedure using the little-JIL process description language. Our goal is an ambitious, vertically integrated software environment in which run-time data sets drive the organization of behavior and contribute to the management of large and comprehensive software systems. This document describes the very first experiments employing this paradigm.

2 Sensory Primitives for Motion Tracking

A multi-objective system requires that the sensory algorithms are flexible to support adaptation and reconfigurable on-line to facilitate fault-tolerance. Our approach is designed to provide a set of sensor processing techniques that can fulfill both low-level and high-level objectives in an open environment. Cooperative interaction among members of the robot team requires the mission planner to be effective in utilizing system resources across team members, including robot platforms, sensors, computation, and communication. In particular, we are constructing virtual robot behaviors across multiple coordinated platforms and multiple sensors. To achieve the desired robustness, our platform is configured with a variety of sensors and algorithms. Vision is the primary sensing modality, but it is complemented by inexpensive pyroelectric sensors, sonar, infrared proximity sensors, and (in the future) acoustic sensors. Multiple types of sensors are considered to be distributed across multiple robot platforms to allow flexibility in mission planning and resource scheduling in response to hardware and algorithm failures.

2.1 Panoramic Imaging

Effective combinations of transduction and image processing is essential for operating in an unpredictable environment and to rapidly focus attention on important activities in the environment. A limited field-of-view (as with standard optics) often causes the camera resource to be blocked when multiple targets are not close together and panning the camera to multiple targets takes time. We employ a camera with a panoramic lens[2] to simultaneously detect and track multiple moving objects in a full 360-degree view [4, 10, 13].

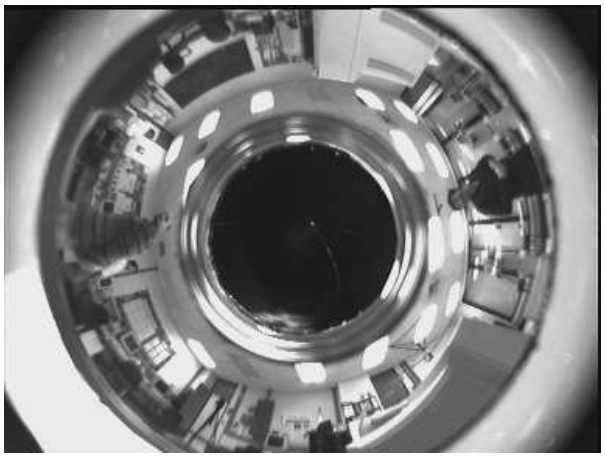


Figure 1. Original panoramic image (768 x 576)

Figures 1, 2, and 3 depict the processing steps involved in detecting and tracking multiple moving humans. Figure 1 shows one of the original panoramic images from a stationary sensor. Four moving objects (people) were detected in real-time while moving in the scene in an unconstrained manner. A background image is generated automatically by tracking dynamic objects through multiple frames. The number of frames needed to completely build the background model depends on the number of moving objects in the scene and their motion. The four moving objects are shown as an un-warped cylindrical image of Figure 2, which is a more natural panoramic representation for user interpretation. Each of the four people were extracted from the complex cluttered background and annotated with a bounding rectangle, a direction, and an estimated distance based on scale from the sensor. The system tracks each object through the image sequence as shown in Figure 3, even in the presence of overlap and occlusion between two people. The dynamic track is represented as an elliptical head and body for the last 30 frames of each person and the final position on the image plane is illustrated in Figure 2. The human subjects reversed directions, overlapped,

[2] PAL-3802 system, manufactured by Optechnology Co.

Figure 2. Un-warped image, four moving people detected

Figure 3. Track through image sequence for the last 32 frames

and occluded on another during this sequence. The vision algorithms can detect self-motion of the robot, change in the environment, illumination, and sensor failure, while refreshing the background accordingly. The detection rate of the current implementation for tracking two objects is about 5Hz.

The motion detection algorithm relies heavily on the accuracy of the background model at any given time in order to detect moving objects. Types of changes in the background can be broadly grouped into two categories.

- changes due to the illumination affecting pixel intensities at a fine scale; and
- changes of surfaces in the environment such as the movement of objects.

It is quite difficult to take care of both cases simultaneously because the first type requires a constant update while the second type requires a context-dependant update. The low-level background estimation procedure is quite simple. The constant update is done on those regions of the image that are not classified as a moving object by the motion detection algorithm. We track each region and keep a history of velocity for each as well. When the velocity falls below a threshold and remains so for a period of time, it becomes a suitable candidate for part of the background. The assumption is made that humans will be not be still for a long period of time. Therefore, they do not become part of the background. Similarly, only when the velocity of an object exceeds a threshold, is it classified as a possible human subject. This helps to avoid detecting some objects that should remain part of background but are not completely stationary, like the motion of tree branches, or the flicker of a computer monitor.

The adaptive background update improved the performance of the panoramic sensors considerably. The above adaptation only provides a low-level mechanism to handle the problem of maintaining an accurate background model. A more elegant way would be to use the context as inferred by the reasoning at higher levels of knowledge-based planning where all resources available might be employed. For example, an unconscious human will be still, so the low level will infer this as the background appearance. However, using the pyroelectric sensor, we might know where the human is, particularly if the previous motion of that body had been detected. This information could be passed to the vision sensors to update the background accordingly.

2.2 Pyroelectric sensor

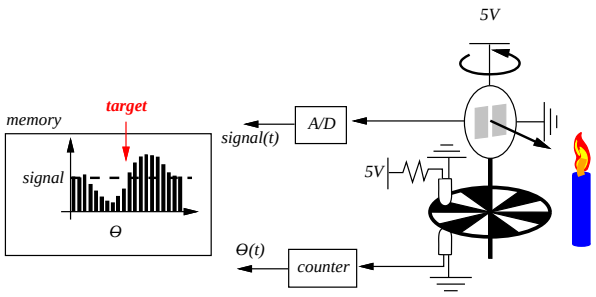


Figure 4. Pyroelectric sensor.

The pyroelectric sensor[3] is a Lithium Tantalate pyroelectric parallel opposed dual element high gain detector with complete integral analog signal processing [3]. The detector is tuned to thermal radiation in the range that is normally emitted by humans. Since the pyroelectric detector itself only responds to changes in heat, the detector

[3]Model 442-3 IR-EYE Integrated Sensor, manufactured by Eltec Instruments

must be scanned. As shown in Figure 4, a thermal target is identified as a zero crossing in the sensor's data stream. We have implemented such a sensor on a scanning servo motor with two control modes; the sensor may saccade to a region of space designated by another sensor or pair of sensors, and it can track the thermal signature (even when the subject is still) by oscillating around the target heading. The result is a sensor that responds quite precisely to human body temperature but with a rather poor lateral bandwidth. This is due primarily to the scanning required to measure a zero crossing. To use this sensor appropriately, it must be applied only when the predicted lateral bandwidth of the subject is relatively small.

2.3 Stereo Head System

The stereo head platform[4] is a high-performance binocular camera platform with two independent vergence axes. As shown in Figure 12, it has four mechanical degrees of freedom and each lens has three optical degrees of freedom [6].

There are several state-of-the-art tracking algorithms in the literature [1, 9, 2]. Our tracking algorithm uses one of the cameras as an active eye and the other as an passive eye. The active eye detects subsampled pixels of greatest change in intensity between two consecutive frames. The passive eye correlates multi-resolution fovea with the frame from the active eye. The stereo head is then servoed to bring the pixel of greatest change into the fovea of the active eye. Subsequently, the passive eye is verged to point its fovea to the same world feature as the fovea of the active eye, extracting the spatial location of the object.

The accuracy of the spatial location of the object is dependent on its distance from the stereo head system. This algorithm can only track single moving objects.

2.4 SACCADE-FOVEATE B-Pgm for Recovering Heading

The most primitive software process in this approach is an asymptotically stable closed-loop *controller* [5, 8]. Controllers suppress local perturbations by virtue of their closed-loop structure. Some variations in the context of a control task are simply suppressed by the action of the controller. Controllers also provide a basis for abstraction. Instead of dealing with a continuous state space, a behavioral scheme need only worry about control activation and convergence events. When a control objective is met, a predicate is asserted in an abstract model of the system behavior. The pattern of boolean predicates over a working set of controllers constitutes a functional state description in which policies can be constructed. The "state" of the system is a vector of such functional predicates, each element of which asserts convergence for some control law and resource combination. The state vector also, therefore, represents the set of discrete subgoals available to a robot given these native control laws and resources.

Two closed-loop primitives are employed for motion tracking (see Figure 5).

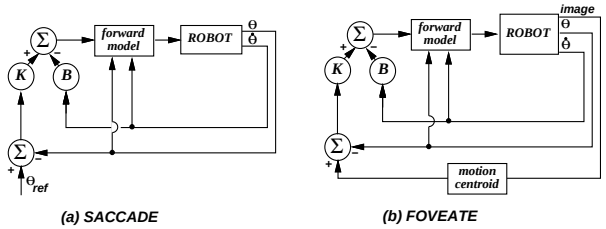


Figure 5. Closed-Loop Primitives for Controlling Attention.

The first, *saccade*, accepts a reference heading in space and directs the sensor's field-of-view to that heading. The second, *foveate*, is similar except that it accepts heading references determined by the sensor's signal. For example, the pyroelectric sensor scans a small region centered on the current gaze and identifies the *zero crossing* in the sensor output. The heading to the zero crossing is used as the reference heading to control the sensor's gaze. Within bandwidth limitations, the result is that the pyroelectric sensor tracks the moving thermal source.

Localizing and tracking the motion of objects in the world is an important, reusable behavior that can be realized a number of different ways using a variety of different sensors. Each sensor in a stereo pair recovers the heading to a feature in the environment. When the imaging geometry of the pair is suitable, the sensors can, in principle, be used to triangulate the spatial location of the feature. Moreover, the control process for each sensor can be completely independent of the other sensor processes. We have handcrafted a B-Pgm for accomplishing this task that is parametric in sensory resources. This B-Pgm is illustrated in Figure 6 - it represents a family of run-time hardware configurations for estimating the location of moving objects in space.

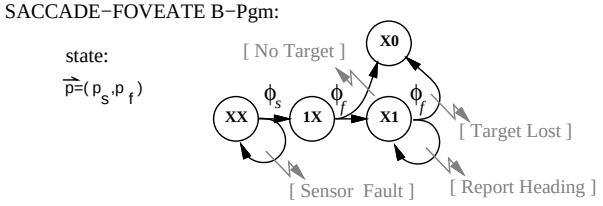


Figure 6. Behavior Program for Detecting and Measuring the Heading to a Motion Cue.

[4] BiSight System, manufactured by HelpMate Robotics, Inc.

The state in the nodes of Figure 6 is the convergence status of the saccade controller, ϕ_s, and the foveate controller, ϕ_f. That is, if ϕ_s is converged and ϕ_f is not, then the state of the saccade-foveate process is 10. An X in the state representation represents a "don't care" or "don't know" condition.

The saccade-foveate B-Pgm (or template) for behavior relies on a concurrent *"saccade-foveate"* approach. This strategy begins by directing a sensor $r_1 \in R$ to saccade to an interesting region of space. If this process fails for some reason, it is presumably an error in the motor component for the sensor and it reports a fault. If no hardware fault is detected and the sensor achieves state $1X$, then an independent, periodic, closed-loop process ϕ_f is engaged whose goal it is to bring the centroid of the local motion cue to the center or fovea of sensor r_1's image plane. If no motion cue is detected, then a report of *"no target"* is generated. If a target motion cue is detected and foveated, then the sensor achieves state $X1$ where the target is foveated, is actively tracked, and which likely is no longer at the position specified by the original saccade. As long as subsequent foveation cycles preserve this state, a heading to the motion cue is reported. If, however, the sensor state becomes $X0$, then the target may be moving too quickly and a *"target lost"* report is generated. When two sensors are simultaneously in state $X1$, then the pair of active B-Pgms are reporting sufficient information for triangulating the spatial location of this motion cue. Under these circumstances, this B-Pgm produces a hypothesis regarding the location of a motion cue. Each unique resource allocation $r_1, r_2 \in R$ produces hypotheses of varying quality depending on the context of the localization query.

This policy does not rely on specific output type. In fact, while incorrect correspondence can lead to anomalous results, cross-modality can be used to advantage. For example, if the location is computed from consistent visual motion and pyroelectric information, then we may detect *"warm-moving"* bodies. Such a strategy may be attractive when detecting and localizing human beings as opposed to other types of moving objects.

2.5 "Virtual" Stereo Pairs

Any fixed-baseline stereo vision system has limited depth resolution due to the imaging geometry, whereas a system that combines multiple views from many stationary or movable platforms allows a policy to take advantage of the current context and goals in selecting viewpoints. A *"virtual stereo"* policy is a policy that engages different sensor pairs as the target moves through ill-conditioned sensor geometries. Although this policy is more flexible than a fixed pair, this approach requires dynamic sensor (re)calibration and accuracy in the depth of a target is limited by the quality of calibration. The virtual stereo strategy may be particularly effective with a pair of mobile panoramic sensors because they have the potential of always seeing each other and estimating calibration parameters[13]. Once calibrated, they can view the environment to estimate the 3D information of moving targets by triangulation, and maintain their calibration during movement by tracking each other. If two panoramic vision sensors can see each other and at the same time see the target motion cue, then they can be used to estimate the bearing and distance of the target without any off-line calibration.

$$D_1 = B\frac{\sin(\beta_{12} - \theta_2)}{\sin(\beta_{12} - \beta_{21} + \theta_1 - \theta_2)} = B\frac{\sin(\alpha_2)}{\sin(\alpha_0)} \quad (1)$$

where D_1 is the distance between the target and the first camera, B is the distance between the two cameras, θ_1 and θ_2 are the bearings of the target in image 1 and image 2 respectively, and β_{12} and β_{21} is the image of camera 1 in image 2, and camera 2 in image 1 respectively. Several practi-

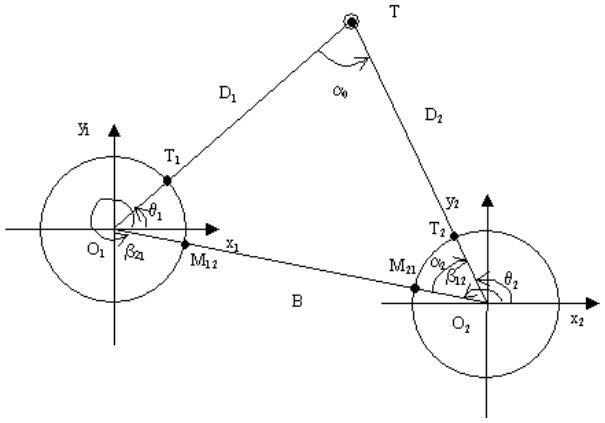


Figure 7. Panoramic stereo geometry

cal approaches to estimate distance and angles between two panoramic sensors have been proposed in [13]. The error of D_1 can be estimated by partial differentials of Equation 1 as

$$\partial D_1 = \left|\frac{\sin(\alpha_2)}{\sin(\alpha_0)}\right| \partial B + B \left|\frac{\sin(\alpha_0 + \alpha_2)}{\sin^2(\alpha_0)}\right| \partial\alpha \quad (2)$$

where ∂B is the distance error, and $\partial\alpha$ is the average angle error. The smaller the angle α_0 and/or B, the larger is the error. Notice that α_0 and B have some inherent dependency. Given the distance D_1 and D_2, the change of α_0 and B are in the same direction (increasing or decreasing).

We have developed the algorithms for mutual calibration and 3D localization of motions using a pair of panoramic vision systems each running the saccade-foveate B-Pgm. The first implementation has been carried out by cooperation between two stationary cameras. Figure 8 shows a stereo image pair from two panoramic sensors.

Figure 8. 3D localization by the panoramic stereo system

2.6 Peripheral and Foveal Vision Integration

The human eye has a very wide-angle low resolution field in its peripheral view and a very high resolution narrow field in its foveal view, a combination that works cooperatively in a highly robust manner. We can find a moving object within the peripheral field of view, and then start a tracking behavior by peripheral-foveal cooperation. The key point here is the natural cooperation of peripheral and foveal vision as a real-time behavior operating within a common coordinate system.

As we consider a computer implementation of this behavior, we note differences with human capability. Humans must rotate the head so that the peripheral system covers the moving object in its field of view. Furthermore, multiple objects in very different directions cannot be tracked simultaneously. In our Track Human Containment Unit, the panoramic-panoramic sensor pair (or any other pair applicable under the run-time context) can provide the spatial reference for a saccade-foveate B-Pgm on a standard zoom camera mounted on a small pan/tilt platform. The pan/tilt/zoom imaging system may then undergo a saccade to the interesting motion cue. From here it can foveate on the cue and zoom if necessary for detailed processing.

High resolution color images obtained from the pan/tilt/zoom camera can be used to determine the identity of the object of interest. In particular, a challenging problem is to separate and track individuals in a group (or even a crowd). Using contour extraction algorithms based on motion cues, the pixels that correspond to the object can be extracted from the background.

Our general approach is to apply suitable local image operators to the image to determine the relevant features of the object. Each known object is represented as a histogram of these local features. The histogram of the object being tracked can be matched with the histograms of other known objects from a database in order to recognize the object. Object recognition is important when there are multiple objects being tracked. When the paths of two moving objects intersect, an ambiguity arises as to whether the paths did, indeed, cross or whether both objects turned back upon meeting. We presented such a situation in Figures 2 and 3.

We have successfully set up a peripheral and fovea vision system, and implemented a cooperative algorithm for processing moving objects. The system detects any moving object in the view of the panoramic camera, and tracks and identifies it through the zoom camera. If there are multiple motion trackers orchestrated in the Human Tracker CU and multiple pan-tilt zoom cameras in a distributed sensor network of stationary and moving platforms, the functionality of the system should respond gracefully in the face of hardware and algorithm failures by deploying applicable subsets of sensors.

Figure 9 illustrates the image resulting from such a process where the spatial reference to a motion cue is provided by the panoramic-panoramic image pair presented earlier in Figure 8. The suspicious character in this panoramic image pair has been scrutinized successfully using the pan/tilt/zoom camera.

Figure 9. A close up (zoom) image of the Human Subject localized using a panoramic-panoramic sensor pair.

3 The Containment Unit

B-Pgms can be used to coordinate the behavior of a fixed set of resources. In [7], we show how to build policies automatically using reinforcement learning that approach optimal policies for a fixed resource allocation. The Containment Unit (CU) is an active entity designed to represent a family of optimal contingency plans parameterized by resource commitments. Its objective is to "contain" faults. A fault is generally construed to be any functional violation of the specified behavior associated with the containment unit: real-time constraints, liveness of constituent hardware, or performance constraints. If a sensor fails, it is the role of the containment unit to select an alternative behavioral program to provide the same type of information and to inform the process that activated the CU of the impact on the expected performance. Containment units, therefore, manage a set of parametric B-Pgms given resource specifications and report the property associated with the CU and the expected quality of the result. The CU monitors fault conditions and responds autonomously to produce the information requested by a higher-level CU.

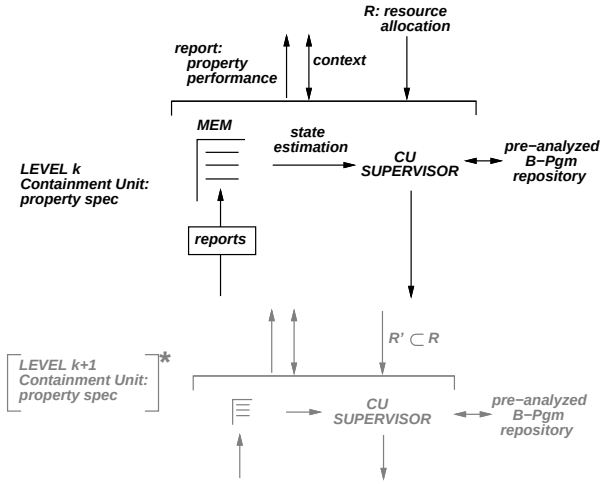


Figure 10. The Structure of a Containment Unit.

The structure of a CU is presented schematically in Figure 10. Multiple instances of a CU may be active concurrently, each with a resource specification that determines the range of variation permitted locally in the strategy for executing the CU directive. Global resource constraints are achieved by limiting the range of autonomy each CU enjoys through careful specification of its proprietary resources. The set of alternative B-Pgms available to the CU represents all possible coordinated sensory and motor policies for achieving the objective with systems resources. In general, these policies may be applicable only in prescribed contexts. For example, adequate illumination may be necessary to employ those B-Pgms with vision sensors, or limited target velocity may be required in order to track with a scanning pyroelectric sensor. These "contexts" can be loaded when a CU is activated and then verified at run-time, or they may be recovered by monitoring the active B-Pgm's performance. An inappropriate run-time context can be used to reconfigure the CU locally and/or passed upward to the process that activated the CU.

3.1 CU Supervisor: Domain-Independent Behavioral Expertise

Some aspects of a particular B-Pgm's performance *in situ* are determined entirely by attributes of the participating resources. The most obvious example of critical local state is the *liveness* of the participating hardware. Other locally determined attributes can also be important with respect to overall performance. Consider a pair of vision sensors performing as a virtual stereo pair to localize a moving target. Localization will be poor if the uncertainty in the position of the participating sensors is large or the saccade-foveate B-Pgm may behave poorly if the target approaches a collinear spatial relationship with the sensor pair. These conditions are entirely determined by examining attributes of the sensors (their relative spatial arrangement) and the result of the B-Pgm coordinating them (the target position).

Circumstances such as these are completely determined in the local state of the CU and should be handled locally without higher-level deliberation. The CU depicted in Figure 10 contains a local supervisor that accomplishes this objective. Some of the policies engaged by the supervisor can be hand-coded based on knowledge regarding the interactions between resources and/or known deficiencies in software processes used to respond to feedback from the world. We will develop an example of the CU supervisor in Section 5.

3.2 Context: Domain-Dependent Behavioral Models

Open environments present data sets to sensorimotor processes that cannot be predicted at process configuration time in general and must be observed at run-time. When peculiar or unexpected environments cause the behavior of the system to deviate from expectations, a higher-level reconfiguration must modify system performance while remaining within specifications. If a specific B-Pgm proves to be inadequate in a particular run-time context, the context is passed upward in the control hierarchy to a process manager which may choose to reinstantiate the CU with a different resource specification. Over time, some of these reconfiguration decisions that depend strongly on controllable system components might be compiled into appropriate CU supervisors. However, other contexts will be determined by the run-time environment, and the deliberative process planner must model these dependencies at a higher

level. We are studying mechanisms where the process description can incrementally model these environmentally determined contexts and manage resources so as to recover critical run-time, environmentally determined contexts in the course of the mission.

In self-adaptive software systems, an additional dimension of complexity in decision-making and predictability is introduced, namely, adaptive changes must be accomplished with some assurance of correctness and safety. Exhaustive formal methods, or detailed re-planning are often too time consuming to be executed completely at run-time. This system exploits a dynamic composability approach, wherein possible execution models are pre-analysed at design time to determine which compositions could potentially execute successfully. The approach also generates an expectation of the successful completion of these compositions under different anticipated contingencies (including hardware faults, algorithm failures, and deviations from expectations). This pre-analysis will greatly reduce the overhead of dynamic decision making, by ruling out alternatives unlikely to be fruitful and by guiding the search process at run-time with this composability information. With this, many of the estimates can be tightened and more efficient plans can be generated. Thus, when a fault is reported, the Containment Unit executes a search over the repository of available B-Pgms and chooses the one whose pre-analized composability information is most appropriate for the task, state of the system, and state of the world.

The memory structure illustrated in Figure 10 records the reported results of all participating resources, estimates the state information required by the local supervisor, and supports interpretation and reporting actions generated by the CU. Task specific information such as target location and current fault conditions are stored. The structure is maintained by a communication protocol over internet sockets between the active B-Pgms and the CU. If resources reside on disparate architectures and operating systems, the memory structure will also provide the CU with a common communication interface to all subsystems. The memory structure is also available to any process running on the host computer and forms the basis for the High Level Interface.

4 The Little-JIL Agent Coordination Language

Little-JIL [LJIL-ICSE] provides rich and rigorous semantics for the precise specification of processes that coordinate multiple agents [11, 12]. In the context of SAFER, the agents consist of individual sensors, individual robots, or combinations of these. Little-JIL provides constructs for proactive and reactive control, exception handling and resource management.

A Little-JIL process defines a high-level plan to coordinate agents to act as a loose team. A process is constructed of steps that are hierarchically decomposed into finer-grained substeps. The steps and substeps are connected with dataflow and control flow edges. Each step may have a resource declaration identifying the resources needed to carry out that step. These resources include the sensors and robots but may also include computational platforms and communication hardware to allow reasoning over the sharing of these resources among computationally and communicationally expensive algorithms.

A process typically specifies parts of the coordination quite precisely while leaving some opportunities for choices to be deferred until runtime. For example, precise resource allocation decisions are typically deferred to runtime (or a pre-runtime analysis stage). In this way a step may be implemented in one of several ways, each of which uses a different collection of resources. The selection of which choice is most appropriate may depend upon which resources are available at that time, how quickly we must perform the computation, how precise a result we must get, and the physical environment at the time of execution. These high-level decisions that require reasoning across the collection of loosely-coupled robots and sensors are the types of decisions made within the process.

The process also contains a reactive element. This is particularly useful for exception handling. For example, a certain amount of reaction can be handled within the containment units by dynamically selecting the appropriate B-programs. Some situations, however, require higher level support. A simple example is that of a timeout. We may want to instantiate a particular containment unit for a limited amount of time. To do this we inform the containment unit of a timeout. When the timeout occurs, the process reacts by choosing another activity based upon the results seen thus far. Another example occurs with a process intended to track multiple people. With such a process, we might want to always have one sensor responsible for watching for new motion entering at a door, while allowing the remaining resources to track targets already in the room. If a new motion enters, the process reacts by reassigning resources. The actual selection of resources and containment units and thus the actual instantiation of the system is made by the integrated capability of robot planning and scheduling technologies whose description is outside the scope of this paper.

The Little-JIL process control language as discussed above, provides a powerful means of exploiting knowledge to structure planning and learning by focusing policy formation on a small set of legal programs. Moreover, at lower levels, new and enhanced processes are constructed. The objective is to constantly optimize and generalize the active B-Pgm during training tasks, and to return it at the end

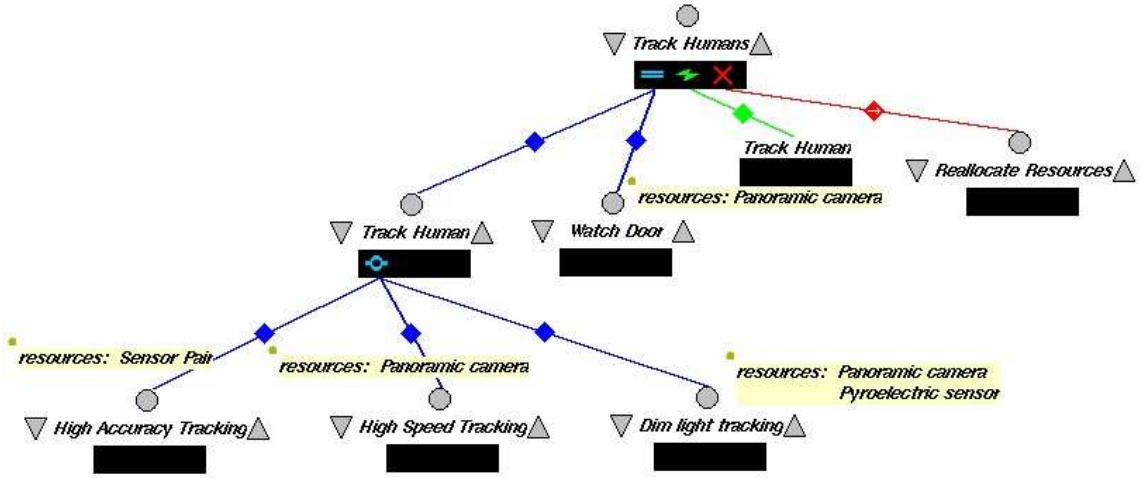


Figure 11. Sample Little-JIL Process Description for Tracking a Human Subject.

of the task better than we found it. These B-Pgms actually consist of many coordinated primitive controllers but are thought of as discrete abstract actions. Subsequent plans and learning processes can exploit this abstraction.

Figure 11 shows a sample Little-JIL process that uses sensors to track multiple humans. We assume that this process specification is in the context of a partial model of the run-time environment. The root step of the process is Track Humans. This step is decomposed into two steps that run concurrently (denoted by the blue parallel lines). One step is to track an individual human while the other step is to watch the door. The Watch Door step requires use of the panoramic camera.

Track Human is a choice step. Dynamically, the system will decide which of the three substeps to use. This decision will be based upon which resources are available, what time constraints there are on the tracking, and contextual issues, such as whether there is good lighting or whether the target is moving quickly. One might easily imagine many more than three choices here. Each choice requires one or more resources and has some expected performance. The scheduler and runtime system use knowledge about the context to assist in making the decision.

If another human enters the room, this results in an event that is handled by the second Track Human step. This is simply a reference to the original track human step and will result in a new instance of Track Human starting with a new set of resources. Of course, when a new human enters the room, it could be that the existing resources are all being used to track the people that are in the room. This would result in an exception causing some replanning and reallocation of resources to occur. Other exceptions can be used to adapt locally (within the CU) during execution. For example, if there had been normal lighting and the lights were turned off, we would expect an exception within the currently active containment units that employ vision sensors.

5 SAFER Experimental Platform

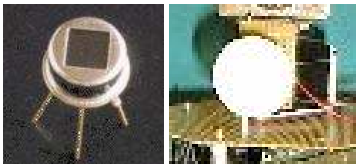
In our experimental platform, we have implemented three types of motion detectors that are deployed at fixed and known positions in an indoor office-like environment. The platform consists of an articulated stereo vision system, and scanning pyroelectric sensor, and two panoramic vision sensors. In each instance of the saccade-foveate B-Pgm observations are collected from sensor pairs that are sufficient to determine a spatial location of the moving feature in the field of view. This family of functionally equivalent programs produces a spatial estimate of a motion cue with varying quality that could serve as a spatial position reference to a subsequent sensory or motor control task. Indeed, combinations of these strategies are themselves B-Pgms with reserved resources for corroboration or for fault tolerance. Which of these to use in a particular context is dependent on the task, the resources available, and the expected performance based on accumulated experience.

5.1 Designing the CU Supervisor for Tracking Human Subjects

The CU Supervisor determines which B-Pgm (sensor pair) is recommended for triangulation and tracking give the current state of the process. In our demonstration, there are six unique pairs of sensors available. A state predicate describes the “liveness” of each pair. For a given pair, if both sensors are functioning and they are not in a collinear

Motion Tracking Sensors:

- 1 Pyroelectric Sensor;
- 1 Stereo Head Sensor;
- 2 Panoramic Vision Sensors.



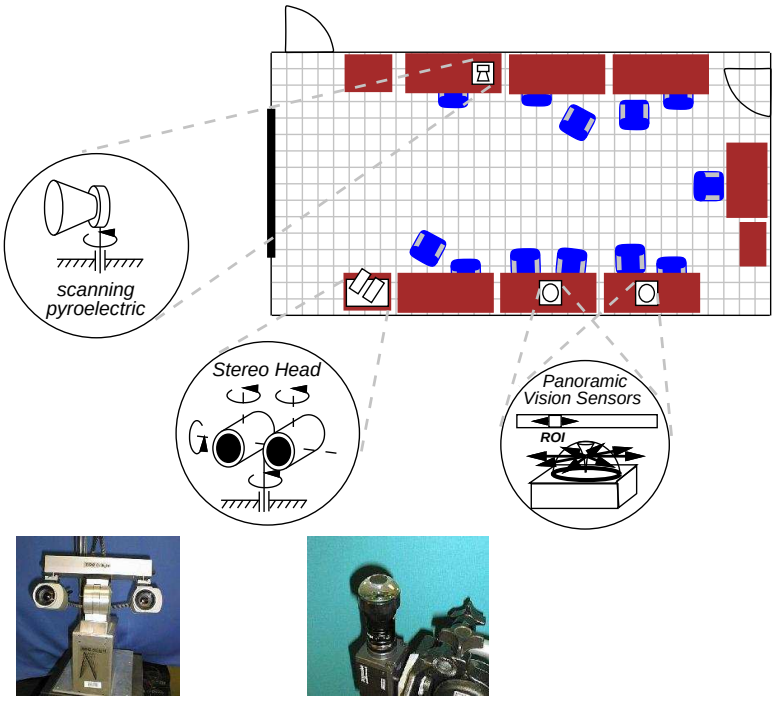


Figure 12. The "Smart Room" - Motion Tracking Platform.

configuration, the corresponding predicate is set to 1, otherwise it is set to 0. This is the role of the state estimation component of Figure 10. Given a pattern in the state vector defining available sensor pairs, the CU supervisor always chooses the pair of sensors with the highest value with respect to the process' objective function.

We have hand-crafted a Human Tracking CU supervisor for engaging sensor pairs that deploys resources in the following priority-based hierarchy:

- Panoramic-stereo head (camera 1);
- Panoramic-stereo head (camera 2);
- Stereo-head (camera 1 and 2);
- Panoramic-pyroelectric;
- Stereo-head (camera 1)-pyroelectric;
- Stereo-head (camera 2)-pyroelectric.

Each resource allocation in this hierarchy, in turn, instantiates two concurrent containment units for tracking motion with a single sensor. These subordinate CUs execute the saccade-foveate B-Pgm described earlier and report to the track human CU. Each CU in this hierarchical control process has the authority to manage the resources reserved for them.

5.2 Experimental results

The Human Tracking CU supervisor has been implemented to control the various sensors in order to track a single moving person seamlessly through failure modes captured in the liveness assertion. Some preliminary results are presented below.

5.2.1 Accuracy and Repeatability Experiments.

To design any CU supervisor that depends on the coordinated activity of multiple sensors, it is necessary to model the performance of the individual sensors. We conducted a series of experiments to determine the accuracy and repeatability of the sensors. At known spatial locations, a motion cue was generated and observed from the different sensors. It was observed that the panoramic sensors were both accurate and repeatable, the stereo head is accurate but not repeatable, and the pyroelectric sensor was repeatable but not accurate. The data was also used to examine the quality of triangulation on the motion cue by different sensor pairs. As expected the quality degraded as the motion cue approached the line joining a sensor pair or a collinear configuration. Because such a configuration is not desirable we call this a collinear fault. Conversely, the quality is best when motion cue is along a direction orthogonal to the line joining a sensor pair.

5.2.2 Tracking a Human Subject.

The next experiment evaluated the complete task of tracking a single moving person using combinations of the four sensors. The results are shown in Figures 13, 14, 15 and 16. Figure 13 shows the tracks of Panoramic-

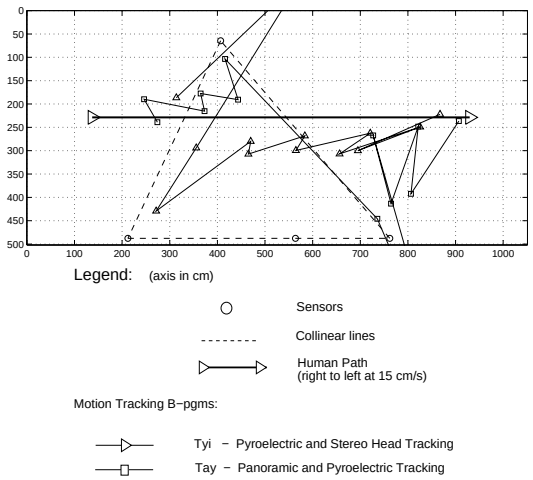


Figure 13. Motion Tracking for the Pyroelectric-Stereo head and Pyroelectric-Panoramic sensor pairs in the "Smart Room.".

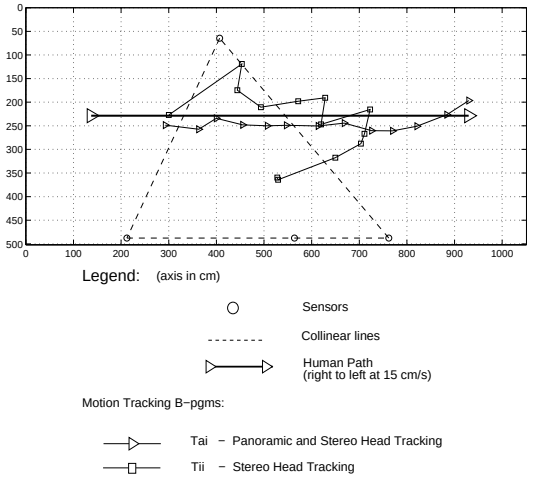


Figure 14. Motion Tracking for the Panoramic-Stereo head and Stereo Head sensor pairs in the "Smart Room.".

Pyroelectric pair (T_{ay}) and Pyroelectric-Stereo head pair (T_{yi}). As the motion track crosses collinear sensor geometries, the performance degrades as expected.

Figure 14 shows the tracks of Panoramic-Stereo head pair (T_{ai}) and Stereo head alone (T_{ii}). Target tracking using stereo head alone can be quite bad due to the small stereo baseline and the mechanical properties of the Stereo Head platform [2].

Figure 15 shows the localization results using the Panoramic virtual stereo pair (T_{ap}) produces very good results for large regions of the room. This sensor pair is therefore highly reliable, leading to its priority in the CU supervisor for the task. When these resources are available, they are well-advised both for tracking precision and because of the complete field of view they provide.

The last example show the performance of the CU supervisor which effects software mode changes in response to liveness feedback from the sensors. This feedback addresses both hardware function and the collinearity fault. Figure 16 shows that the Track Human CU supervisor was effective in handling these run-time contexts. The observed track was very to the true trajectory of the moving test subject.

The results demonstrate that the hierarchical architecture is capable of handling faults at both lower level (i.e. sensors) and higher level (i.e. context of the motion cue).

6 Summary, Conclusions, and Future Experimental Work

Multi-robot scenarios present significant technical challenges regarding sensing, planning, computing, and software methods and must support both reactivity and predictability. Ultimately, one of the most desirable characteristics of a multi-robot system is its ability to adapt to changes in the environment and to internal faults - in hardware components and in end-to-end performance specifications. Thus, reconfigurability is critical.

Our current work presents some preliminary results towards the responsiveness to novel data sets, adaptability and robustness that are critical to a multi-robot application. The CU supervisor that was assigned the task of tracking a human was able to handle individual sensor faults (low-level) as well as faults due to context of the motion-cue (high-level) and seamlessly track the human. In conclusion, our vertically integrated software environment can reconfigure resources dynamically depending on a variety of failures, making the system robust.

6.1 Doorway Abstraction

In a real situation, the agents will have to cooperate with each other and pool their sensory information and knowledge to build a reliable model of their environment. To demonstrate one such situation, we plan to give the agents the task of doorway abstraction. The doorway is an interesting thing to learn and incorporate it into the model because that is were motion cues of interest originate often. A panoramic sensor could be useful in this task because its

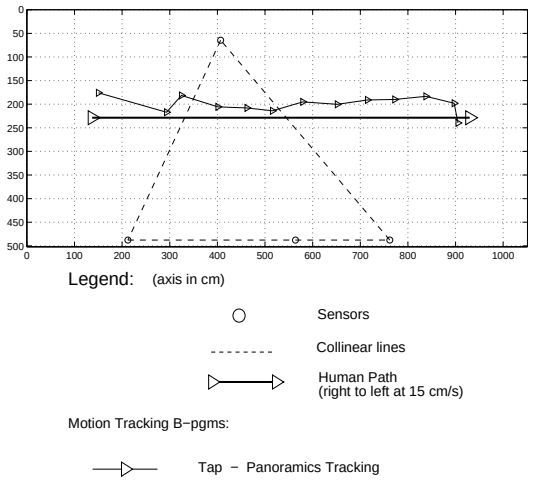


Figure 15. Motion Tracking for the Panoramic-Panoramic sensor pair in the "Smart Room.".

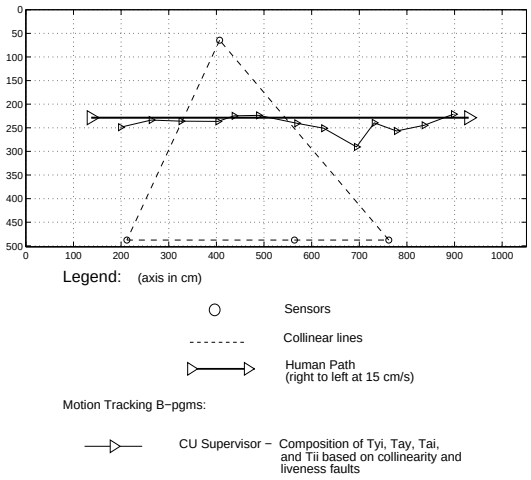


Figure 16. Motion Tracking Performance during Mode Changes in the Motion Tracking CU supervisor..

field of view is 360 degrees unlike other sensors which has to saccade to a region of interest. The panoramic sensor can maintain a certainty value for different regions in its field of view. The value indicates the certainty of a doorway being in that region. Whenever a motion cue appears or disappears in a region, its certainty value is increased. This way after certain period of time, a reliable model about doorway could be built. Of course, some of these regions need not be doorways but just simple occlusions. This can be handled by using two panoramic sensors, one of which is moving and can position itself from where it can corroborate the presence or absence of doorway.

6.2 Multiple Target Corroboration

When there are multiple targets, the triangulation is no more trivial. The different types of sensors come to our aid is this situation. Like for example if panoramic sensor is tracking two motion cues - say a person and robot, using the pyroelectric sensor we can know that one of them is a person and thus selectively track that motion of more interest to us. Another scenario is when two motions cues intersect. This problem is discussed earlier in Section 2.1. When the scenario involves multiple humans, it is more challenging. In this case, we plan to use a monocular camera to zoom in on regions of interest as given by the panoramic sensor, as shown in Section 2.6. This helps us to capture and record signatures of these motion cues like color, shape etc. The higher levels can reason over these and help to decide which targets to triangulate on.

References

[1] David Coombs and Christopher Brown. Real-time binocular smooth pursuit. *International Journal of Computer Vision*, 11(2):147–164, 1993.

[2] Konstantinos Daniilidis, Ch. Krauss, Michael Hansen, and Gerald Sommer. Real time tracking of moving objects with an active camera. *Real-Time Imaging*, 4(1):3–20, February 1998.

[3] Eltec Instruments, Daytona Beach, FL. *Eltec Pyroelectric Sensor*. Distributed by Acroname.

[4] P. Greguss. *Panoramic Imaging Block for three-dimensional space*, January 1986. U.S. Patent 4566763.

[5] Roderic A. Grupen, Manfred Huber, Jefferson A. Coelho Jr., and Kamal Souccar. Distributed control representation for manipulation tasks. *IEEE Expert*, 10(2):9–14, April 1995.

[6] HelpMate Robotics Inc., Danbury, CT. *BiSight System*.

[7] M. Huber and R. A. Grupen. A feedback control structure for on-line learning tasks. *Robotics and Autonomous Systems*, 22(3-4):303–315, December 1997.

[8] M. Huber, W. S. MacDonald, and R. A. Grupen. A Control Basis for Multilegged Walking. In *Proceedings of the International Conference on Robotics and*

Automation, volume 4, pages 2988–2993, Minneapolis, MN, April 1996.

[9] Atsuto Maki, Tomas Uhlin, and Jan-Olof Eklundh. Phase-based disparity estimation in binocular tracking. In K. Heia, K. A. Høogdra, and B. Braathen, editors, *Proceedings of the 8th Scandinavian Conference on Image Analysis*, pages 1145–1152, Tromsøo, Norway, May 1993. Norwegian Society for Image Processing and Pattern Recognition.

[10] S. K. Nayar and S. Baker. Catadioptric image formation. In *Proceedings of DARPA Image Understanding Workshop*, pages 1431–1437, May 1997.

[11] A. Wise. Little-JIL 1.0 Language Report. Technical Report TR 98-24, University of Massachusetts Amherst, Dept. of computer Science, February 1998.

[12] A. Wise, B. S. Lerner, E. K. McCall, L. J. Osterweil, and Jr. S. M. Sutton. Specifying coordination in processes using little-jil. Technical Report TR 99-71, University of Massachusetts Amherst, Dept. of Computer Science, 1999. Submitted to ICSE 2000.

[13] Z. Zhu, E. M. Riseman, and A. R. Hanson. Geometrical Modeling and Real-Time Vision Applications of Panoramic Annular Lens (PAL) Camera Systems. Technical Report TR 99-11, University of Massachusetts Amherst, Dept. of Computer Science, February 1999.